# Appendix A. Short case descriptions

This appendix introduces the five cases that we have studied. The cases are independently introduced, to give the reader the possibility of getting an impression of the firms and of the variety present in these cases. In the appendix, we describe the characteristics of the cases. In the main text of Chapter Two, we pay attention to the types of relationships between cells and the corresponding co-ordination requirements. The objective of the case studies is to identify the variety of planning problems of firms that apply cellular manufacturing in their small batch mechanical part production.

The description of the cases is organized as follows. After a short introduction of the company, we characterize the situation in terms of the type of products, demand characteristics, requirements of the market, size of the assortment, number of employees and number of shifts.

Next, the cellular organization is described, accompanied with a scheme of the goods flow. In these schemes, cells are located at a specific processing stage. We distinguish between cells (represented as *boxes*), departments of similar cells (*dotted rectangle*), and service departments that are shared by the cells (*dotted ellipses*). External service departments or subcontractors are positioned outside the boundaries of the production system. *Arrows* represent the flow of goods in the system. *Triangles* represent stock positions, and *dotted triangles* mark that these positions are not in constant use.

We further pay attention to the degree of automation, type of layout, criterion used in the formation of the cells, presence of a remaining cell, and the sequence of visiting the cells if a main goods flow in the system exists.

Finally, characteristics of the available resources are described. We pay attention to the storage and registration of tools and the corresponding investment policy. Furthermore, we address the operator flexibility and other instruments, such as overtime and subcontracting, which are used to obtain the required flexibility in the production system.

At the end of this appendix, we summarize several relevant aspects of the cases we studied. We focus on the number and type of cells in these firms.

The information on these cases is gathered from interviews during short visits and meetings with employees of these firms in 1994 and 1995, and from reports of student research projects that we supervised in these firms. In the description of the cases, the anonymity of the firms is respected as much as possible.

## A.I. Case I Complex machines

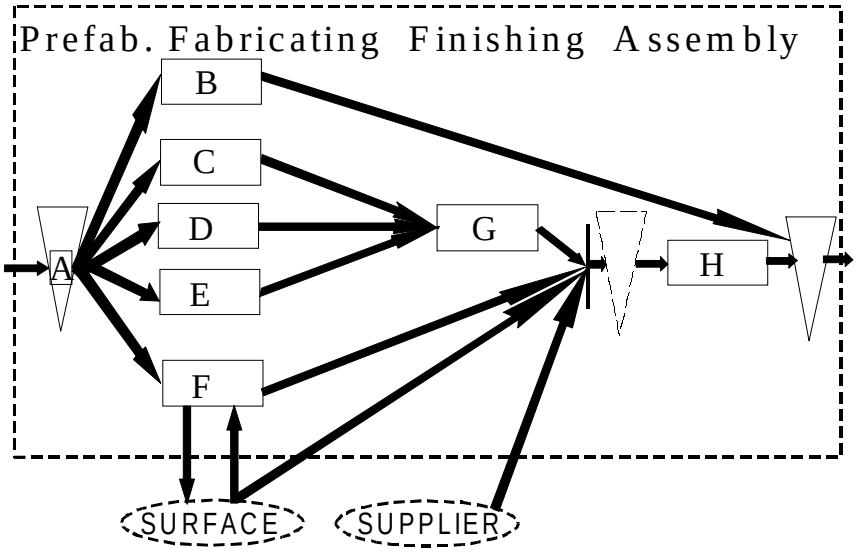



**Figure A.1 Goods flow case 1 Complex machines**

Case I is producing complex machines that are an important element of the capital intensive production process of their industrial customers, 95% of them being situated in the Far East. The machines are being made to order, with some engineering activities for specific requirements of a customer. The lead time of a machine is half a year. In the production site that we have studied, work 550 people in a two shift production.
The processing stages that were present are material processing, mechanical parts production, finishing and assembly. The required sheet metal is produced in another production site and either delivered to the assembly stock or to a parts producing cell. Half of the 100,000 parts and components are being bought externally. Complementary to the new machines, spare parts are demanded. These parts are responsible for 10% of the machine hours.

The production system consists of a prefabrication cell A, a remaining cell B, four parts production cells C-F, a finishing cell G, and an assembly cell H, as can be seen in Figure A.1. Two of the parts producing cells only perform machining operations (C and E); the other two can also perform operations on sheet metal (D) or simple assembly operations (F). The degree of automation varies between the cells. C is highly automated, D and F use various NC machines, and E is hardly automated.

The cells were designed by using the criterion ‘shape and volume of the products’, which explains the various degrees of automation. The change to cellular manufacturing was initiated by the introduction of an FMS system.

The remaining cell B has specialized on the fabrication of tools. This cell is not involved in the main production activities due to the high costs of the people who work in this cell.

Tools that are exclusively used in one cell are stored decentrally. They can be placed in a separate toolmagazine, but cutting tools can also be placed in an integrated toolmagazine within the machine. Tool types that are used by more cells are being duplicated as much as economically can be justified, so no resource conflicts will arise. Still a considerable number of tools exist (especially fixtures) that are centrally stored and released on request. The resulting flows are not controlled.

Due to the presence of similar operations in the various cells, some operator flexibility is available between the parts producing cells. Allocation of work orders is done independent of the current work load balance between the cells. If a cell is overloaded, it can self decide about the interchange of work with either another cell or an external subcontractor.

A considerable percentage of the parts have to undergo one or more finishing operations. Special surface operations are subcontracted, taking 2 weeks throughput time, but cleaning and painting are done in the finishing cell G. Three colors are used for painting, one of them irregular. The inventory before assembly is a decoupling stock; it functions as a time buffer.

## A.II. Case II Complete installations

Case II is producing complete installations, consisting of a number of highly automated complex machines that are placed in line. Due to the modular design of the installations, a high degree of standardization within the production is achieved, especially in the fabrication of the machines, which is done in the production site we visited. The installations are being made to order with some engineering activities for specific requirements of a customer. The firm produces ± 35 installations a year, cumulating to a demand of 750 machines a year, demand of spare and wear parts being complementary. Of the 35,000 part types used in these machines, 15,000 are produced internally. Production is done in two shifts and some 400 people work in this organization.

The requirements on quality, reliability and lead time performance are very strict, as the installations are a vital element of a continuous production process. The replacement of the old installation with the new one is planned a few months in advance and has to take place within a very short time. These requirements have led to the introduction of cellular manufacturing.
The cellular organization in this site consists of a prefabrication cell A, five parts production cells C-G, one remaining cell B, one finishing cell H and three assembly cells I-K (Figure A.2). The required elements for the conveyor chain are produced in another production site and delivered to the assembly stock.

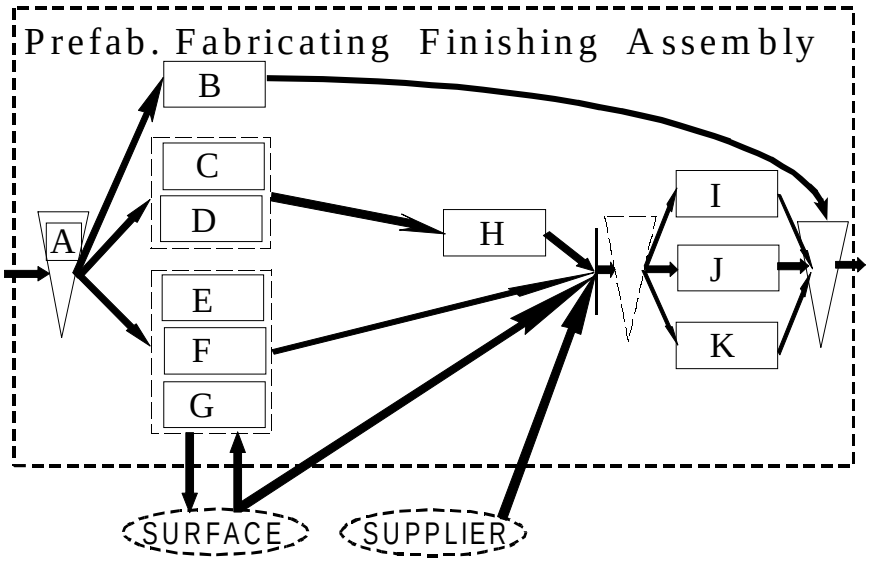



**Figure A.2 Goods flow case II Complete installations**

The parts production can be divided in a cluster of three cells E-G where the mechanical parts are being produced, and a cluster of two cells C-D for sheet metal operations. The latter is related to the finishing cell. The degree of automation in the mechanical parts production cluster is very high, as the simple work is structurally subcontracted. One of the sheet metal cells (D) also performs some simple machining operations on the material.

The remaining cell B has specialized on the fabrication of tools and prototypes. This cell is only involved in the main production process if high precision repair work or rush work has to be done. External subcontractors are often cheaper, so their involvement is generally preferred.

The parts production cells are designed by using the criterion ‘shape of the products’ and orders are allocated to the cells based on the required main processing operation. Assembly cells are not specialized in specific products. The inventory before assembly is a decoupling stock with a time buffer of minimal three working days.

The tools that are needed in parts production are stored decentrally. A tendency exists towards designing more cell-specific tools in stead of product-specific tools. Capacity in the assembly cells is partly determined by the availability of product carriers.

Operator flexibility within the clusters of machining cells E-G and sheet metal cells C-D is much higher than between these clusters. Cells can also choose to work in overtime making use of a machine in another cell. The cell foreman is allowed to take subcontracting decisions on his own. He stays responsible for the subcontracted work, for the delivery of the material, tools, NC programs, and documents, and for the lead time performance.

## A.III. Case III Complex installation

Case III produces complex installations that consist of two main modules, Y and Z. A considerable part of the demand is on module Y only, which is delivered in four basic types, each in ten variants obtainable. Module Y is assembled to order. 80% of the orders for module Z require some engineering activities before production can start. The throughput time of an installation is eight weeks, due to the late start of the production of Z if engineering activities are necessary. There are 300 employees; half of them are directly involved in the production.

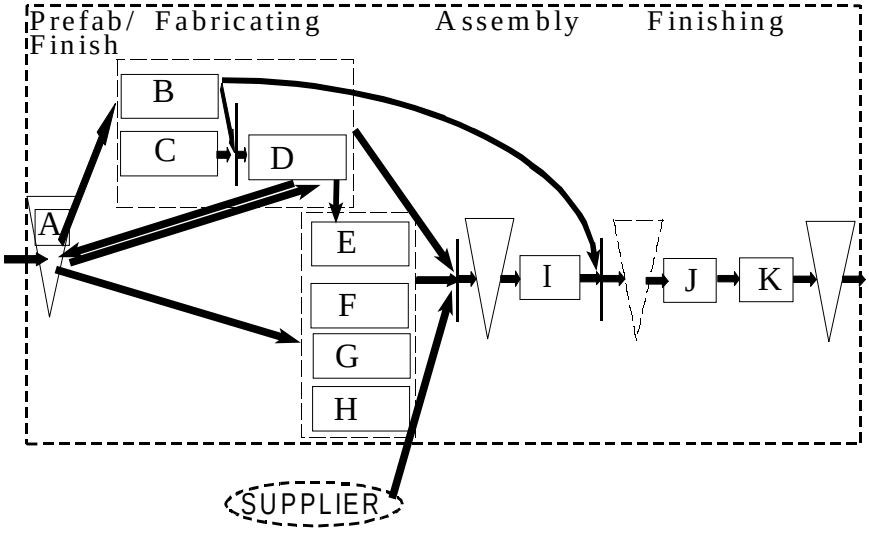



**Figure A.3 Goods flow case III Complex installation**

Production takes place in cells (see Figure A.3). The cellular organization consists of one pre/post processing cell (A), three welding cells (B-D), a cluster of four machining cells (E-H), one of them being a remaining cell (H), two assembly cell (I, J) and a finishing cell (K). Before the assembly cells a planned stock is found.

The prefabrication cell delivers the required material to the parts production cells. Cell B produces module Z and delivers it to assembly cell J. In the production of module Y first the welding cluster is involved, next the machining cluster, and afterwards the main part is put in stock. In the welding cluster first cell B is visited. Cell D uses the parts produced in the highly automated cell C and welds them on the part delivered by cell B. Cell D uses a line layout for this welding operation. Next, for the required postprocessing operation cell A is (again) visited. The quality check is again performed in cell D.

The next step in the production of module Y is machining in the highly automated cell E. This cell produces in two manned and one unmanned shifts. The other three cells in the machining cluster produce parts for the assembly of module Y. One of them (H) is a remaining cell that produces tools, performs maintenance, and can be involved for rush work. Cell F and G are dedicated to specific part families; G is even using a line layout.

Cell I performs the assembly of module Y. Cell J is involved if a complete installation is ordered and cell K performs the painting of the modules and installation. Tools that are used by more cells are duplicated and decentrally stored. Transportation equipment for intracell transport is a shared resource.

Much of the required flexibility is found in capacity inventory and in operator flexibility. A human resource pool is available in both clusters, although the pool is larger in the welding cluster. Reallocation of members of this pool is considered on request of a cell. In the machining cells, some simple work can be interchanged. Subcontracting is not preferred, although sometimes inevitable.

## A.IV. Case IV Parts production make/engineer to order

Case IV produces parts. It is directly connected with its two main customers that belong to the same parent organization and are responsible for 90% of the orders, both new and repetitive. 10% of the orders require engineering activities. A direct linkage with the CAD systems of the main customers supports a quick delivery of the parts requested. The lead time of the orders ranges from one to four weeks. The organization employs 300 people, of which 190 are involved in the fabrication of parts, which is done in two shifts. Some 6000 different part types are produced each year.

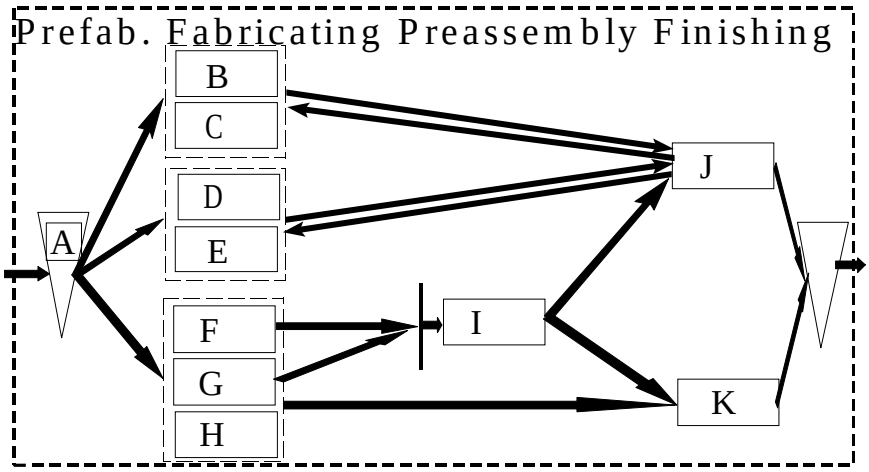



**Figure A.4 Goods flow case IV Parts production make/engineer to order**

Production is done in eleven cells (see Figure A.4): one pre-processing cell (A), two machining cells (B, C), three sheet metal cells (D-F), two punching cells (G, H), a pre-assembly cell (I) and finally two finishing cells (J, K). There is no remaining cell, as the fabrication of tools for both the main customers and themselves is performed in a separate part of the organization. The formation of the cells is based on the criterion ‘shape of the product’ and each cell is equipped with both numerically controlled and conventional

machines. One of the sheet metal cells is highly automated and can produce during an unmanned shift.

At the time the cellular organization had been adopted in this firm (1987), it consisted of eighteen cells divided over the three clusters. Cells within a cluster were more alike with respect to the type of machines allocated to these cells, and this would make the interchange of work and operators easily. Since then, the number of cells has been reduced due to diminishing turnover and problems with making use of the available flexibility between the cells.

Tools are central stored, even when the tools are product specific and used in one cell only. Central storage provides more space on the work floor and is preferred. Some of the tools that are used in more cells are duplicated, but this is regarded as a too expensive solution. Information on the availability and location of the tools is not registered.

The operator flexibility within the clusters is high, but not often used. Subcontracting for capacity reasons is possible, but not preferred. A small number of part types are sometimes used for producing capacity inventory, but the involved risk is often too big. Use of overtime can be decided on by the cell. If a cell delivers work to another cell, it stays responsible for the quality and delivery performance.

## A.V. Case V Parts production make to order

Case V produces mechanical parts for all business units of its parent organization. It produces 6000 orders a year and has an assortment of 20,000 parts that can be made to order. The lead time ranges from five to seven weeks. Production is done in two shifts and 100 people are working in this organization.

The cellular organization (see Figure A.5) consists of a pre-processing cell (A), four machining cells (B-E), E being a remaining cell, a sheet metal cell (F), and a finishing cell (G). Machining cell B is a highly automated cell with a flow line layout. Furthermore, two separate departments (H, I) are shared by the cells. In department H, the operator performs surface operations, based on a quick service concept. To department I no operators are allocated. The numerically controlled machines in this department can be used by any cell. Each cell has operators that can handle these machines. The remaining cell E is involved in the production of prototypes and tools and is not often used for rush work or interchange of operators.

The two machining cells C and D produce parts for different product/market combinations, i.e. for different business units. The types of machines that are allocated to these cells are alike, but as far as the numerically controlled machines concerns not interchangeable, due to

the different programming languages. The machinery consists of 125 conventional machines and almost 40 CNC machines.

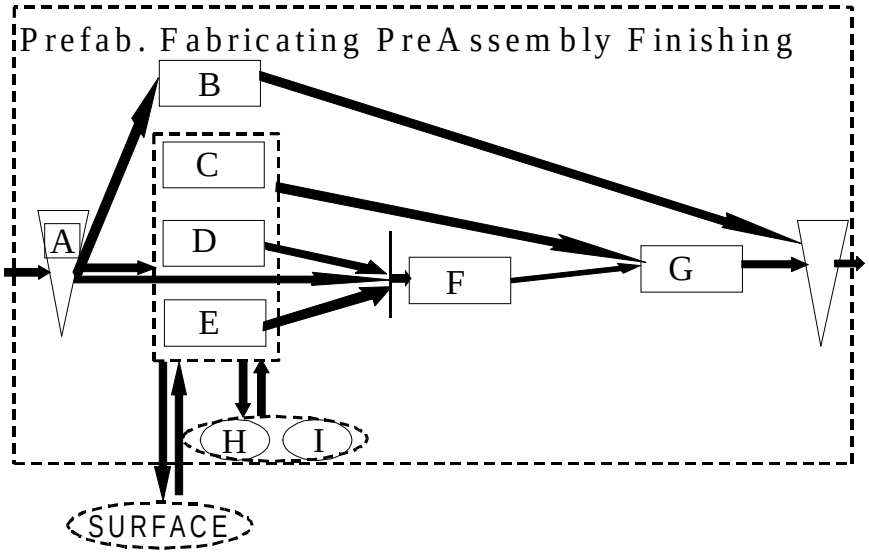



**Figure A.5 Goods flow case V Parts production make to order**

As much as 20% of the orders require operations outside the cell. In this percentage, the operations performed in departments H and I are not included. Most of these operations are externally performed surface operations. An example of internally performed operations concerns orders, allocated to the sheet metal cell, that require welding of parts. These parts first have to be made in other cells, causing operations to be performed outside the cell.

The firm has the disposal of 6,000 different standard tools and 20,000 types of special tools. Tools are central stored. Duplication of tools is often too expensive. Location and usage of tools is not registered. Some 5% of the orders are delayed because the required tools are not available at the time production has to use it.

Operator flexibility is not of main interest to this firm. The required system flexibility is found in changing the allocation of simple work between the machining cells, in delaying some work, or in using overtime, subcontracting or temporary workers in the finishing department.

# A. SUMMARY OF CASE DESCRIPTIONS

The case descriptions show a variety of cellular configurations, a different degree of automation between cells, and various contributions of a remaining cell in generating flexibility in the primary process. Table A.1 summarizes the main characteristics of these five cases in terms of number and type of cells.

| Cases | I | II | III | IV | V |
|---|---|---|---|---|---|
| end product | complex machines | complete installations | complex installations | parts | parts |
| production situation | make/engineer to order | make to order | assemble to order | make/engineer to order | make to order |
| assortment | 50000 parts | 35000 parts | ? | 6000 parts/year | 20000 parts 6000 parts/year |
| Type of Cell | Number of Cells | | | | |
| *prefabrication* | 1 | 1 | 1 | 1 | 1 |
| *mechanical* | 2 | 3 | 3 | 5 | 3 |
| *plate/welding* | 0 | 1 | 3 | 5 | 1 |
| *combined plate/ mechanical* | 2 | 1 | 0 | 0 | 0 |
| *finishing* | 1 | 1 | 1 | 2 | 1 |
| *assembly* | 1 | 3 | 2 | 1 | 0 |
| *remaining cell* | tool fabrication | tool fabrication | tools + specials | NA | tools+prototypes |

**Table A.1 Characteristics of cellular manufacturing systems in case studies**

# Appendix B. Mixed Integer Programming for Stage Allocation

In order to facilitate the allocation of operations to stages, we have developed a mixed integer programming model that explicates the factors that should be taken into account when determining the contents of the stages. These factors have been discussed in Section § 4.4.

Define

N = Number of stages
T = Manufacturing throughput time
P = Period length P ≡ T/N
$j$ = Index of stage ($j$ = 1..N)
$h$ = Index of product ($h$ = 1..H)
$i_h, l_h$ = Index of operation $(i_h, l_h = 1..n_h)$
$\overline{d_h}$ = Mean demand for product $h$ in next N sales periods of length P
O = Set of operations to be performed during T
$F_i$ = Set of immediate followers of an operation $i$
s(i) = Setup time for operation $i$
p(i) = Processing time for one unit of operation $i$
MI = Machine Interference delay as percentage of period length P
$Ta_j$ = Tardiness in stage $j$
$I_{i_h l_h}$ = 1 if operations $i_h \in$ O and $l_h \in F_{i_h}$ are performed in different cells
CD = Cell delay (incurred if $I_{i_h l_h} = 1$ and both operations are allocated to the same stage)
$V_{i_h}$ = Value added after one unit of operation $i_h$ has been made (including value of purchased parts)
$C_T$ = Inventory cost as percentage of value of one stocked item during T time units
M = Big number (M >> P)

Decision variables

$x_{i_h j}$ = $1 \Leftrightarrow$ operation $i_h$ allocated to stage $j$; otherwise $x_{i_h j} = 0$
$t_{i_h j}$ = earliest start time of operation $i_h$ in stage $j$

Having introduced the nomenclature we use, we present the *longest-path orientation* of the model in Section B.I and discuss the model. In Section B.II we extend the model with a *bottleneck orientation* that may be useful in designing a PBC system. Finally, Section B.III presents the results of the model for the data of production situation I of Chapter Six.

## B.I. Longest-path orientation in mixed integer programming model

The mixed integer programming model allocates operations to stages such that the investment in working capital is minimized (objective function (1)).

*Model*

$$(1)\quad min \sum_{j=1}^{N}\sum_{h=1}^{H}\sum_{i_h=1}^{n_h} (N+1-j)\cdot\frac{C_T}{N}\cdot V_{i_h}\cdot\overline{d_h}\cdot x_{i_h j} + M\cdot Ta_j \qquad \text{such that}$$

$$(2)\quad \sum_{j=1}^{N} x_{i_h j} = 1 \qquad \forall i_h \in O, h = 1..H$$

$$(3)\quad \sum_{r=1}^{j} (x_{i_h r} - x_{l_h r}) \geq 0 \qquad \forall i_h \in O, l_h \in F_{i_h}, h = 1..H, j = 1..N-1$$

$$(4)\quad t_{l_h j} + M\cdot \sum_{r=j+1}^{N} x_{l_h r} \geq t_{i_h j} + s(i_h) + \overline{d_h}\cdot p(i_h) + I_{i_h l_h}\cdot CD - M\cdot \sum_{r=1}^{j-1} x_{i_h r} \qquad \forall i_h \in O, l_h \in F_{i_h}, h = 1..H, j = 1..N$$

$$(5)\quad t_{i_h j} + s(i_h) + \overline{d_h}\cdot p(i_h) \leq (1-MI)\cdot P + Ta_j \qquad \forall i_h = 1..n_h, h = 1..H, j = 1..N$$

$$(6)\quad x_{i_h j} \in \{0,1\}, t_{i_h j} \geq 0, Ta_j \geq 0 \qquad \forall i_h = 1..n_h, h = 1..H, j = 1..N$$

An operation $i_h$ that is allocated to stage 1 will add $C_T \cdot V_{i_h} \cdot \overline{d_h}$ to the total holding cost.

If this operation had been allocated to stage N, the added cost would have been $\frac{(C_T \cdot V_{i_h} \cdot \overline{d_h})}{N}$

However, the allocation of operations to stages is restricted for several reasons.

First, we define the *occurrence-constraint*. An operation has to be allocated to at least one stage in order to let the production system perform the operation. Constraint (2) together with condition (6) that $x_{ij}$ is a binary variable ensures that the operation has to be allocated to one and only one stage. The uniqueness of stage allocation is a consequence of PBC.

Precedence relationships between operations make that if an operation *i* is allocated to stage *j*, then none of its followers *l* may be allocated to an earlier stage. Constraint (3) ensures this *precedence-feasibility* of the solution.

The reason why not all operations can be allocated to the final stage N is that the allocation has to be *time-feasible* as well. If too many sequentially dependent operations are allocated to the same stage, this may cause tardiness to occur. Constraints 4-5 determine the tardiness in a stage due to the sequential relationships between operations in the same stage, and the objective function minimizes the tardiness. The non-negative variable $t_{i_h j}$ describes the earliest starting time of an operation $i_h$ in stage *j*.

Constraint 4 sets the earliest starting time of an operation *l* with a predecessor *i* in the same stage to be not smaller than the earliest starting time of this operation *i* plus the total amount of set-up time and processing time required for the whole batch of *i*. If both operations are not performed in the same cell, then the earliest starting time of the latter operation *l* is further delayed with *CD*, a cell delay (policy) factor, which represents the organizational impact of such an allocation. If *i* and *l* are not allocated to the same stage *j*, then the use of the big *M* factor causes the constraints 4 to become non-binding.

Constraint 5 ensures that the earliest finishing time of any operation in stage *j* will not exceed a specified percentage of the period length P. This percentage depends on *MI*, the machine interference delay, which is expected to correct for delays that may occur in processing a sequence of operations due to the waiting times on availability of machines. If more time is needed in a stage, then the tardiness is assumed to become positive, which increases costs. The objective function tries to minimize costs and hence a solution without tardiness will be preferred. In this way, we are able to model the time/cost trade-off.

Note that the formulas of Chapter Five, Section § 5.1 can also be used to formulate Constraint 4 and recognize the influence of extra transfer batches and duplicate machines. This makes the model far more complex to interpret, and due to some non-linearity's also more complex to solve with standard software. However, the constraints can easily be modified to cope with these factors. This is important, as the solution that is found with our model may not be the most cost-efficient allocation of operations to stages, as we neglect the possibility of shortening the time delays between successive operations.

The actual occurrence of tardiness in a stage depends on the number of products that have to be produced. The sensitivity of an increase of this amount compared with the expected amount $d_h$ may be quite high and depends on the sum of the processing times per unit on such a path in the stage. Instead of the expected demand we could also use the maximum allowable number of units of product *h* per period of length P. This will restrict the occurrence of tardiness problems (time), but will increase the investment (costs).

## B.II. Bottleneck orientation in mixed integer programming model

The former section has described a mathematical model that pays attention to the *longest-path orientation*. It has not considered machine capacity as a dominant factor in allocating operations to stages. The allocation of operations to stages does not influence the amount of work per machine, but it does influence the timing of the earliest arrival at and latest departure from a machine (the start/finish effect). We can add a *bottleneck-orientation* to the model in order to cope with this effect.

A *bottleneck-orientation* has to be applied if the utilization of a bottleneck resource is a dominant design principle in PBC system design. A bottleneck-orientation pays attention to the loading of resources during a period. Our bottleneck-orientation to stage allocation consists of the inclusion of constraints that try to determine if the bottleneck will exceed its capacity limits. The extra constraints and variables for the mixed integer model are presented below. Note that bottleneck problems can be solved with specific methods from single machine scheduling with release dates and due dates. Carlier (1982) developed an algorithm that finds a solution to such problems in O($n$ log $n$) time ($n$ is the number of operations to be scheduled at the machine). The inclusion of these constraints in our mixed integer model serves the purpose of presenting a suitable mathematical formulation of the stage allocation problem. Practical implementations of the model should better use a tailor made algorithm that allows solving much bigger allocation problems within reasonable time.

Constraint 7 ensures that the time delays in stage $j$ are zero if the operation is not performed within stage $j$. Constraint 8 determines the remaining time in stage $j$ after finishing operation $i_h$ at the bottleneck. If the operations $i_h$ and $l_h \in Fi_h$ are performed in different stages, the remaining time is zero, but if they are performed in the same stage, then we have a positive remaining time. This restricts the scheduling capabilities on the bottleneck. Constraints 9-11 pay attention to this problem. For any subset of operations that have to be performed at the bottleneck we determine the earliest release time, add the total set-up time and processing time of this subset at the bottleneck, and finally add the minimal remaining time in this period (at other machines) after finishing this subset at the bottleneck. This sum may never exceed the period length P. All variables are non-negative (12).

Define

| | |
|---|---|
| $B_k$ | = Set of operations $i_h$ that have to be processed at bottleneck machine $k$ |
| K | = Subset of operations: $K \subseteq B_k$ |
| $f_{i_h j}$ | = minimal remaining time in stage $j$ after finishing operation $i_h$ |
| $MinHead_k$ | = minimal waiting time before bottleneck can start with subset K of operations |
| $MinTail_k$ | = minimal remaining time after finishing subset K of operations at bottleneck |

Additional constraints for Model

(7) $$f_{i_h j} \le M \cdot x_{i_h j} \qquad \forall i_h \in B_k, j = 1..N$$

(8) $$f_{i_h j} + M \cdot \sum_{r=1}^{j-1} x_{i_h r} \ge f_{l_h j} + s(l_h) + \overline{d_h} \cdot p(l_h) + I_{i_h l_h} \cdot CD - M \cdot \sum_{r=j+1}^{N} x_{l_h r} \qquad \forall i_h \in B_k, l_h \in F_{i_h}, j = 1..N$$

$$\left\{ \begin{array}{lll} (9) & MinHead_k \ge t_{i_h j} & \forall i_h \in K, j = 1..N \\ (10) & MinTail_k \ge f_{i_h j} & \forall i_h \in K, j = 1..N \\ (11) & MinHead_k + \sum_{i_h \in K} \left\{ s(i_h) + \overline{d_h} \cdot p(i_h) \right\} + MinTail_k \le P & \end{array} \right\} \qquad \forall K \subseteq B_k$$

(12) $$MinHead_k \ge 0, MinTail_k \ge 0, f_{i_h j} \ge 0 \qquad \forall i_h \in B_k, j = 1..N, K \subseteq B_k$$

The number of constraints increases rapidly as the cardinality of $B_k$ increases. However, we need not check every subset $K \subseteq B_k$. We may restrict our attention to subsets of operations that may cause *MinHead* and/or *MinTail* to be non-zero, as this might result in infeasibility with respect to bottleneck capacity. This reduces the number of subsets to be considered.

Due to the specific structure of the sparse matrix of the extended mixed integer model (a large part of it has a structure similar to an uni-modular matrix) an optimal solution is easily found. We have used the Super Lingo software from Lindo Systems Inc. (LINGO manual, 1989) to test the extended model. The limits of this version are a maximum of 1000 variables and 500 constraints. On a Pentium II 266 personal computer, it took less then 2 seconds CPU time to solve problems with approximately this number of variables and constraints.

Note that if products have an identical product structure, but different set-up or processing times, a simplified allocation can be performed with this model by applying the same allocation of operations to stages for all these similar products.

Allocation of operations to stages is an important phase in designing the PBC system. It influences a number of system objectives, such as quality, costs and dependability. The mixed integer model explicates the decisions on stage allocation with respect to the time/cost trade-off. The basic model uses a *longest-path orientation* in the allocation of operations to stages. The model can easily be extended to facilitate a *bottleneck orientation*.

## B.III. Application of mixed integer model on production situation I

In Chapter Six, we perform a simulation analysis using data from the work of Steele, Berry, and Chapman (1995). The stage allocation that we apply is also deduced from their work, and is shown in Figure 6.11 in Section § 6.2.3. We have not used the stage allocation procedure described in this appendix for the allocation of operations to stages in the simulated production situation. At the time of performing these experiments, the limitations of the available LINGO software version were such that we could not obtain a solution for all stages.

We have solved a simplification of the mixed integer model with bottleneck machine 13. The simplification is that all products apply the same allocation of operations to the stages. The optimal stage allocation of this simplified model is presented in Figure B.1.

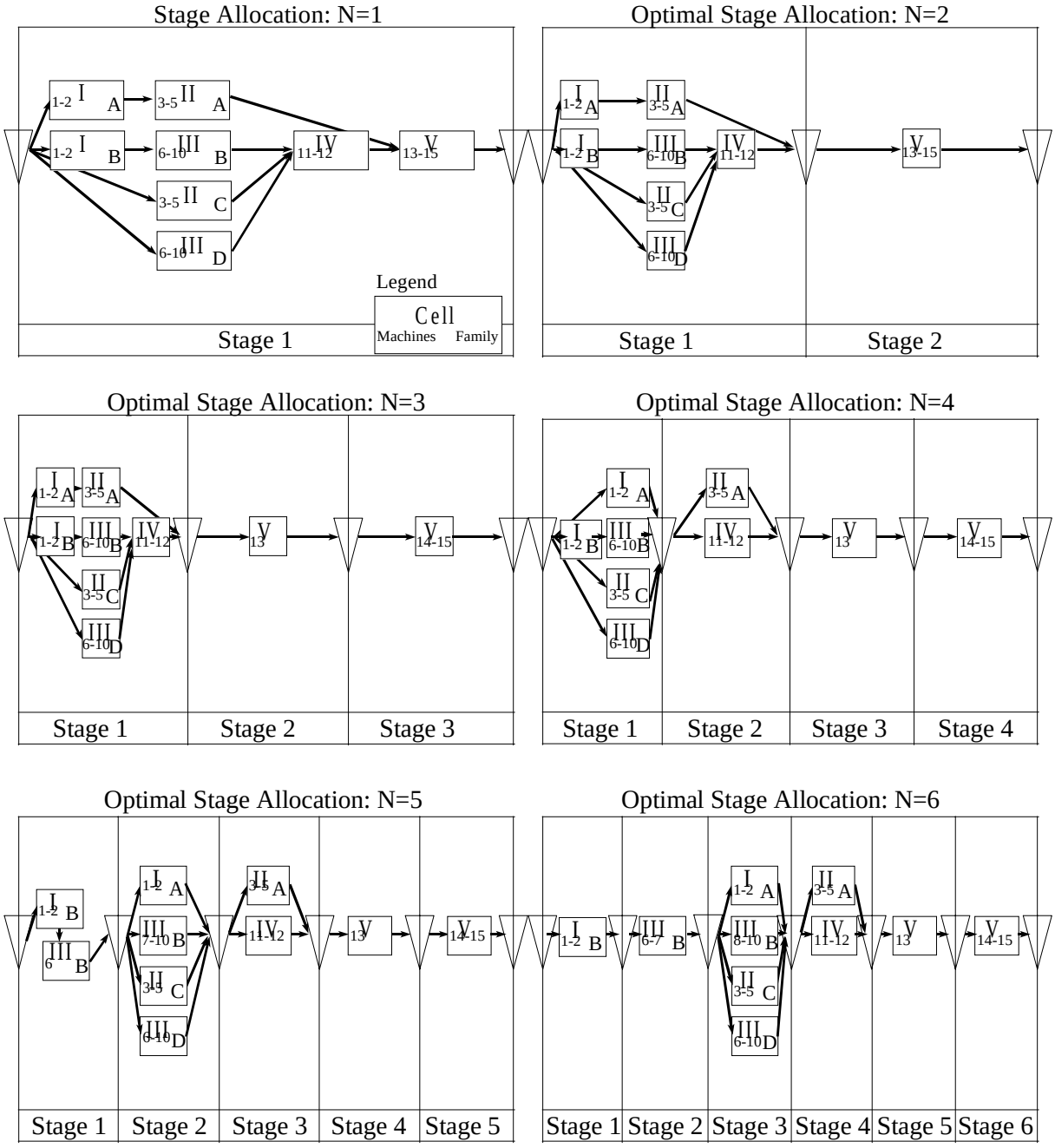



**Figure B.1 Optimal stage allocation for production situation I (machine 13 bottleneck)**

# Appendix C. Proof of equivalence between Formula 5.2 and 5.3

If overlapping production is not applied for a product ($nb_h$=1), the Formula's 2 and 3 of Chapter Five are equivalent.

$$nb_h = 1 \Rightarrow \max_{i=1}^{n_j^h}\left[ r_{hi} + p_{hi} \cdot \left\lceil \frac{q_h}{m_{hi}} \right\rceil^+ + \sum_{t=i+1}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht} \cdot 1} \right\rceil^+ \right\} \right] = \sum_{i=1}^{n_j^h} \left\{ d_{hi} + p_{hi} \cdot \left\lceil \frac{q_h}{m_{hi}} \right\rceil^+ \right\}$$

Proof:

$$\max_{i=1}^{n_j^h}\left[ r_{hi} + p_{hi} \cdot \left\lceil \frac{q_h}{m_{hi}} \right\rceil^+ + \sum_{t=i+1}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht} \cdot 1} \right\rceil^+ \right\} \right] =$$

$$max\left( r_{h1} + \sum_{t=1}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\}, \max_{i=2}^{n_j^h}\left[ r_{hi} + \sum_{t=i}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\} \right] \right) =$$

$$max\left( \begin{array}{l} s_{h1} + \sum_{t=1}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\}, max\left( s_{h2}, r_{h1} + p_{h1} \cdot \left\lceil \frac{q_h}{m_{h1}} \right\rceil^+ \right) + \sum_{t=2}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\}, \\ \max_{i=3}^{n_j^h}\left[ r_{hi} + \sum_{t=i}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\} \right] \end{array} \right) =$$

$$max\left( \begin{array}{l} s_{h1} + \sum_{t=1}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\}, max\left( s_{h2} - p_{h1} \cdot \left\lceil \frac{q_h}{m_{h1}} \right\rceil^+, s_{h1} \right) + \sum_{t=1}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\}, \\ \max_{i=3}^{n_j^h}\left[ r_{hi} + \sum_{t=i}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\} \right] \end{array} \right) =$$

$$max\left( \begin{array}{l} s_{h1} + \sum_{t=1}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\}, s_{h1} + max\left( s_{h2} - p_{h1} \cdot \left\lceil \frac{q_h}{m_{h1}} \right\rceil^+ - s_{h1}, 0 \right) + \sum_{t=1}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\}, \\ \max_{i=3}^{n_j^h}\left[ r_{hi} + \sum_{t=i}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\} \right] \end{array} \right) =$$

$$max\left( d_{h1} + \sum_{t=1}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\}, d_{h1} + d_{h2} + \sum_{t=1}^{n_j^h} \left\{ p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\}, \ldots, \left[ \sum_{t=1}^{n_j^h} \left\{ d_{ht} + p_{ht} \cdot \left\lceil \frac{q_h}{m_{ht}} \right\rceil^+ \right\} \right] \right) =$$

$$\sum_{i=1}^{n_j^h} \left\{ d_{hi} + p_{hi} \cdot \left\lceil \frac{q_h}{m_{hi} \cdot 1} \right\rceil^+ \right\} \qquad \text{as } d_{ht} \geq 0 \qquad \forall t = 1..n_j^h$$

# Appendix D. Exploring related problems with subbatches

In Chapter Five, Section § 5.4.3, we introduce a mathematical model that determines a period length P and a variable subbatch strategy in order to minimize the sum of holding costs, set-up costs and the costs of transfer of subbatches. Various branches of literature have paid attention to the problem of determining batch sizes. The modelling approaches for overlapping operations are all based on the initial work of Szendrovits (1975). The work of Muckstadt and Roundy (see e.g. Muckstadt and Roundy, 1993) gives attention to setting lot sizes in a series of production stages. We discuss a selection of the literature that provided us insight in the characteristics of a suitable solution approach for our mathematical model.

## D.I. Repetitive lots

Splitting a larger batch in equal sized subbatches results in repetitive lots at the various machines. Szendrovits (1975) was one of the firsts who addressed the effect of introducing subbatches on the traditional trade-off between cost factors in the economic production quantity theory. Graves and Kostreva (1986) developed an interesting procedure to determine the optimal number of subbatches. They use the ratio between the holding costs for in-process inventory and the transfer costs as an indicator, as shown in the following expression:

$$nb = \frac{Q}{\sqrt{r}} \cdot \sqrt{\frac{hc}{tc}} \qquad \text{(Graves and Kostreva (1986))}$$

| | |
|---|---|
| $nb$ | number of subbatches |
| $Q$ | optimal economic process batch quantity |
| $r$ | production rate per day |
| $hc$ | holding costs for one unit WIP per day |
| $tc$ | transfer costs of one subbatch |

The results of Graves and Kostreva are worthwhile to consider in developing a solution approach for our model, although their cost structure is not identical to ours.

Kropp and Smunt (1990) experimented with both unequal and equal sized subbatches in a deterministic flow shop for which optimal schedules could be generated and concluded that equal sized subbatches tend to be optimal or near optimal with respect to make span performance in the majority of the test problems. It is important to consider the efficiency of equal sized subbatches in our model in order to see if these conclusions hold also with respect to minimal total cost performance.

## D.II. Overlapping operations in a flow shop

The deterministic flow shop has often been subject of examination for job splitting. Long before Szendrovits developed his EOQ extension, Mitten (1958) had pointed to the use of overlapped production in a flow shop. He modelled the start of the batch at the next machine with start-to-start and finish-to-finish time lags (see Mitten, 1958) and generalised the results of Johnson (1954) through the introduction of start-start time lags in his classical algorithm on solving the two machine flow shop scheduling problem. For a further description of the use of time lags in flow shop scheduling research we refer to Riezebos, Gaalman, and Gupta (1995) and Riezebos and Gaalman (1998). Research on time lag modelling has not paid attention to the determination of the optimal time lag size, which would be useful in determining the number of subbatches.

## D.III. Lot streaming

The lot streaming problem for m machines (operations) is to find the optimal size of a fixed number of subbatches in the system such that a regular measure of performance, e.g. make span or flow time, is minimized. In the lot streaming problem, the number of subbatches is given and constant for each machine. Furthermore, the number of items per subbatch may vary, but the subbatches remain consistent on all subsequent machines, i.e. during processing at the various machines the subbatch is not further split up.

Baker and Pyke (1990) state that for the general lot streaming problem there is no known method of finding optimal solutions, but there do exist solution approaches that find optimal solutions for problems of restricted size (e.g., m-machine, 2-subbatch problem; 2-machine, n-subbatch problem), analogous to results of general flow shop scheduling of Johnson (1954). The size of the first subbatch affects the start time at successive machines, and the size of the last subbatch affects the finish time of these machines. The optimal size of the subbatches is a function of the processing times and the sequence of processing at the machines. Potts and Baker (1989) show that the classical minimal make span flow shop scheduling results with respect to the optimality of permutation scheduling in case of four machines can be transformed to the lot streaming problem. Glass, Gupta and Potts (1992) prove the same for the case of three machines and other regular measures of performance. The use of equal sized subbatches will be optimal in these situations. We can generalize this to the statement that identical subbatch sizes will be optimal on both the first two and the last two operations of a job, due to the invertibility property known from general flow shop scheduling.

Trietsch (1989) applied a maximum on the transfer costs caused by the number of subbatches and minimized the make span with respect to this constraint. Note that our model does just the opposite: we minimize the total cost (including the transfer cost and the holding cost) with respect to a minimum period length, which can be transformed to a maximum make span.

# Appendix E. Progressive search heuristic

The *progressive search heuristic* finds an approximate solution for the mathematical model of Chapter Five. This model decides on a period length P and a variable subbatch strategy $nb_{hi}$ and tries to minimize the resulting total costs. The required number of stages N has a strong impact on total inventory costs. The principles behind this heuristic are described in Section § 5.5.3.

Details of the progressive search heuristic are described in Section E.1 and Section E.2 describes its application on the data of the example problem of Section § 5.5.1. We present the results of the various steps in order to provide more insight in the way the heuristic works. Note that the progressive search heuristic consists of eight steps, including steps for initialization and output generation.

## E.I. Detailed description of progressive search heuristic

STEP 1 INITIALIZE

$\mathrm{P}_{\min} \Rightarrow \max_{k=1}^{K} \left\lceil \dfrac{\sum_{h=1}^{H} \sum_{i^h \in k} s_{hi}}{1 - \sum_{h=1}^{H} \sum_{i^h \in k} \dfrac{p_{hi} \cdot D_h}{m_{hi}}} \right\rceil$ {capacity lowerbound on bottleneck}

$\mathrm{P} \Rightarrow \dfrac{\mathrm{P}_{\min}}{(100 - \mathrm{MI})\%}$ {aim for a bottleneck utilization ≤ (100 - MI)%}

$N_{up} \Rightarrow 0$

$nb_{hi} \Rightarrow 1 \quad \forall h = 1..H, i = 1..n_h$

$\mathrm{Totcost}[j] \Rightarrow \infty \quad \forall j = 1..\infty$

STEP 2 Determine N

$N \Rightarrow N_{up}$

$N_{up} \Rightarrow \max_{h=1}^{H} \left\lceil \dfrac{TT_h}{\mathrm{P}} \right\rceil^{+}$ {largest number of stages necessary with length P}

STEP 3 DETERMINE $P_{(N_{up})}$

$$P_{(N_{up})} \Rightarrow \sqrt{\frac{\sum_{h=1}^{H}\sum_{i=1}^{n_h}(s_{hi}\cdot SC_{hi}+TC_{h'i})}{N_{up}\cdot\sum_{h=1}^{H}D_h * HC_h}} \qquad \{nb_{hi}=1 \ \forall h=1..H, i=1..n_h\}$$

$$P \Rightarrow max\left[P_{min}, P_{(N_{up})}\right] \qquad \{\text{check capacity lowerbound}\}$$

STEP 4 Determine Cost of $(P, N_{up})$

*IF* $N \neq N_{up}$ *THEN* RETURN TO STEP 2 { note that $TT_h$ is a funtion of P, so if P changes we have to iterate until N does not change anymore }

*ELSE* $P_{solution}[N_{up}] \Rightarrow P$

$$Totcost[N_{up}] \Rightarrow P\cdot\left(N_{up}\cdot\sum_{h=1}^{H}\{D_h\cdot HC_h\}\right)+\frac{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\{s_{hi}\cdot SC_{hi}+TC'_{hi}\}}{P}$$

STEP 5 WHILE ($N > 1$) AND ($Totcost[N+1] \geq Totcost[N]$) DO

{

Determine P as Economic Order Interval with known $N$ and $nb_{hi}$

$N \Rightarrow N-1$

$$P_{(N,nb)} \Rightarrow \sqrt{\frac{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\{s_{hi}\cdot SC_{hi}+TC'_{hi}+TC''_{hi}\cdot(nb_{hi}-1)\}}{N\cdot\sum_{h=1}^{H}\{D_h\cdot HC_h\}}}$$

$P \Rightarrow max[P_{min}, P_{(N,nb)}]$

*IF* $\max_{h=1}^{H}[TT_h] > N\cdot P$ *THEN* GOTO STEP 6 { enumerative search heuristic with known P, $N$, and initial subbatches $nb_{hi}$ }

*ELSE* $P_{solution}[N] \Rightarrow P$

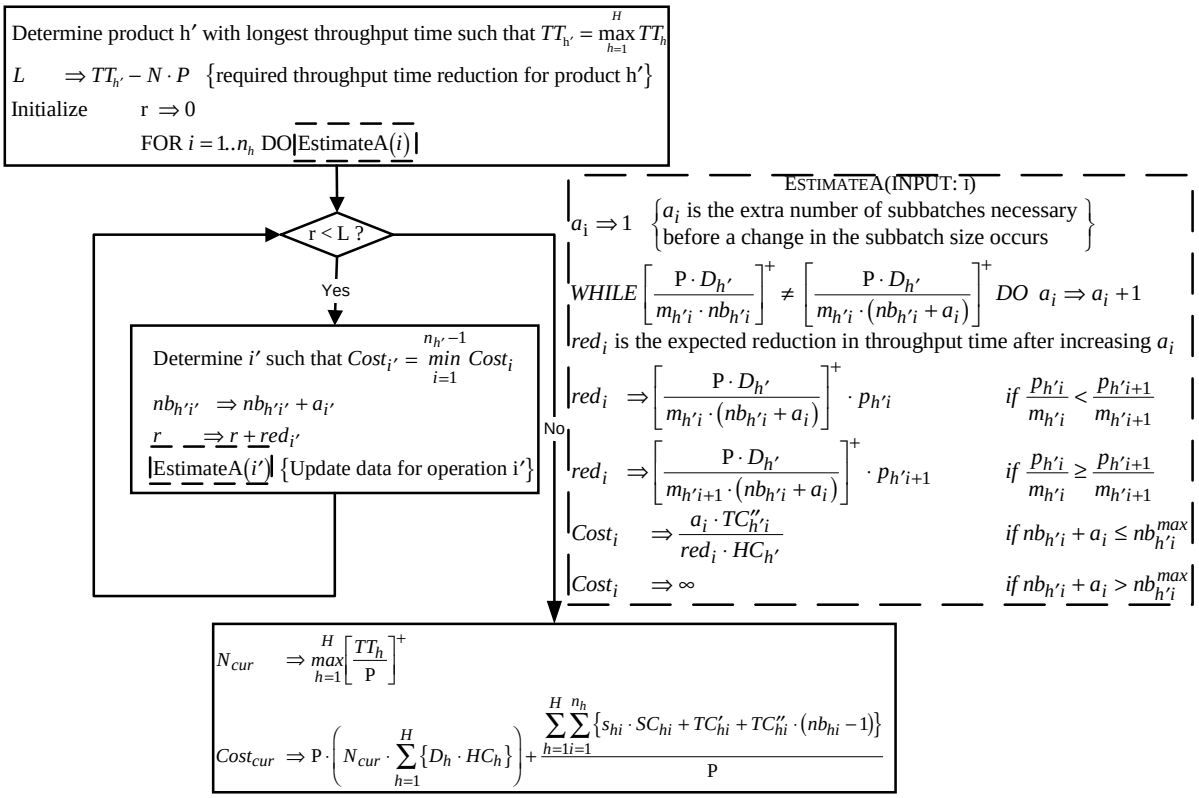
$$Totcost[N] \Rightarrow P\cdot\left(N\cdot\sum_{h=1}^{H}\{D_h\cdot HC_h\}\right)+\frac{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\{s_{hi}\cdot SC_{hi}+TC'_{hi}+TC''_{hi}\cdot(nb_{hi}-1)\}}{P}$$

RETURN TO START OF STEP 5 UNTIL WHILE LOOP ENDS

}

GOTO STEP 8

STEP 6 Enumerative search with known P, $N$, and initial value for $nb_{hi}$




STEP 7 UPDATE SOLUTION

*IF* $\text{Totcost}[N_{cur}] > Cost_{cur}$ *THEN* $\text{Totcost}[N_{cur}] \Rightarrow Cost_{cur}$

$P_{solution}[N_{cur}] \Rightarrow P$

$$P \Rightarrow max\left[P_{min}, P_{(N,nb)} = \sqrt{\frac{\sum_{h=1}^{H}\sum_{i=1}^{n_h} \{s_{hi} \cdot SC_{hi} + TC'_{hi} + TC''hi \cdot (nb_{hi} - 1)\}}{N \cdot \sum_{h=1}^{H} \{D_h \cdot HC_h\}}}\right]$$

$$IF\ \{N_{cur} > N\} AND \left\{Cost_{cur} - (N_{cur} - N) \cdot \sum_{h=1}^{H} \{D_h \cdot HC_h\} < \text{Totcost}[N_{cur}]\right\}$$

*THEN* RETURN TO STEP 6 {re - iterate to further reduce throughput time}

*ELSE* RETURN TO STEP 5 {feasible solution obtained for N stages}

STEP 8 OUTPUT OF *PROGRESSIVE SEARCH HEURISTIC*

$$\text{Totcost}[N^*] \Rightarrow \min_{N=1}^{N_{up}} \text{Totcost}[N]$$

$$P^* \Rightarrow P_{solution}[N^*]$$

## E.II. Application of progressive search heuristic on example problem

Step 1: $P_{min}$ = 0.01442 year (=30 hours)
Step 2: $N_{up}$ = 6;
Step 3: $P_{(Nup, nb=1)}$ = 0.01672
$P = P_{(Nup, nb=1)}$
Step 4: Iterating step 2 and 3 results in the same $N_{up}$
$P_{solution}[N_{up}]$ = 0.01672
$TotCost[N_{up}]$ = 1477
$nb[N_{up}]$ = (1,1,1,1,1,1,1,1,1; 1,1,1,1,1,1,1,1,1)
Step 5: N = 5
$P_{5, nb[Nup]}$ = 0.01832
Step 6: L = 0.00214 (h*=1)
$a_i$ = 1 $\forall i=1..n_1$
$red_i$ = 0.004807 $\forall i=1..n_1$
$cost_i$ = 20.8 $\forall i=1..n_1$
nb = (2,1,1,1,1,1,1,1,1; 1,1,1,1,1,1,1,1,1)
Step 7: TotCost[5] = 1370 $P_{5, nb}$ = 0.01862
Step 5: N = 4
$P_{4, nb[5]}$ = 0.02081
Step 6: nb = (2,2,2,2,1,1,1,1,1; 2,2,2,2,1,1,1,1,1)
Step 7: TotCost[4] = 1353 $P_{4, nb}$ = 0.02298
Step 5: N = 3
$P_{3, nb[4]}$ = 0.02654
Step 6: nb = (2,2,2,2,2,2,2,2,1; 2,2,2,2,2,2,2,2,1)
Step 7: TotCost[3] = 1273 $P_{3, nb}$ = 0.02883
Step 5: N = 2
$P_{2, nb[3]}$ = 0.03531
Step 6: nb = (3,3,3,3,3,2,2,2,1; 3,3,3,3,3,2,2,2,1) $P_{2, nb}$ = 0.03897
nb = (3,3,3,3,3,3,3,2,1; 3,3,3,3,3,3,3,3,1) $P_{2, nb}$ = 0.04034
nb = (4,3,3,3,3,3,3,2,1; 4,3,3,3,3,3,3,3,1) $P_{2, nb}$ = 0.04134
nb = (4,3,3,3,3,3,3,3,1; 4,4,4,3,3,3,3,3,1) $P_{2, nb}$ = 0.04231
nb = (4,4,3,3,3,3,3,3,1; 4,4,4,4,3,3,3,3,1)
Step 7: TotCost[2] = 1255 $P_{2, nb}$ = 0.04263
Step 5: N = 1
$P_{1, nb[2]}$ = 0.06029
Step 6: .... (18 full iterations)
nb = (11,11,11,10,10,10,10,10,1; 14,14,14,12,12,12,12,1)
Step 7: Totcost[1] = 1491 $P_{1, nb}$ = 0.10132
Step 8: Totcost[N*] = min{1491;1255;1273;1353;1370;1477]= 1255
N* = 2
NB* = (4,4,3,3,3,3,3,3,1; 4,4,4,4,3,3,3,1) P* = 0.04263

# Appendix F. Verification and validation of the simulation model

Verification and validation are important steps in the design of a simulation model. We will discuss both steps in the Sections F.I respectively F.II.

## F.I. Verification of the simulation model

In order to verify the simulation model, we have applied various verification methods. First, the structure of the simulation program represents the different processes that were distinguished. We have defined the following processes:

- a customer order arrival process
- a process that releases orders
- a process that generates production orders by exploding the Bill Of Material and that distributes these orders to the stages according to the stage allocation
- a process that controls the progress of each production order in a stage
- a process that monitors information for the stages and overtime control
- a process that represents the activities of a cell
- a process that represents the activities required to process a part
- a main process that controls the simulation run

The contents of these processes are networks of events. If an event is scheduled on the time axis, the activities that are mentioned will be performed. The use of processes facilitates model verification, as it is easy to compare the list of events in the processes with the verbal and schematic description in § 6.2.3 and Figure 6.10.

Another verification method that we applied is the use of debugging within Borland Pascal 7. This allowed us to process the simulation program step-by-step and determine programming errors. As a result of this verification analysis, we have reprogrammed specific error-sensitive procedures in DESIMP that we used to select orders for processing at machines.

We also applied verification methods that are available within DESIMP. The most important verification tool is the *Trace* that describes the events and activities that are being performed in chronological order. We have added relevant information for verification purposes to the standard contents of the *Trace* information. This extra information could be used to control the correct working of the reprogrammed order selection procedures. One of the main problems in our simulation program was the synchronization of several processes at one moment in time. Synchronization at a specific moment in time requires control of the sequence in which specific procedures are called, in order to guarantee that all activities are registered adequately and transferred to the next stage. Using the *Trace,* we were able to control the correct working of this synchronization.

The order arrival and release processes have been verified using animation. This allowed us to control if the release decision was correctly performed.

Finally, we have verified the model for various combinations of input parameters by generating random values for these parameters and analysing the behaviour of the simulated system during these experiments. DESIMP generates *Event lists*, reports the contents of *Queues* and updates results, such as utilization, overtime, and throughput times, during the experiment.

From our verification analysis, we conclude that the model behaves as we expect it to behave.

## F.II. Validation

It is important to pay attention to validation of the simulation study. However, this is not easily to accomplish in our case, as we are not able to compare the outcomes of the model with the real system. We have applied several validation tools in order to facilitate the correctness of the outcomes.

First, we have analysed the outcomes of the simulation studies in several ways. The quantitative output that is generated by DESIMP provides insight in both the mean values and the extremes of the performance measures and other output variables. Analysing the extreme value shows if outcomes are reported that will never occur in practice. However, note that we sometimes are interested in the behaviour of the system in situations that probably never will be tested in practice, because of the high costs of such policies. Hence, extreme outcomes in itself are not suspect, but analysing these extreme values may lead to the conclusion that there are errors in the model.

Another way of analysing the outcomes of the simulation model was to translate these outcomes to a Gantt chart for each period. The use of a different presentation language (graphical representation of the outcomes of the PBC system in terms of schedule per period) helps to clarify if the outcomes are as expected. Figure F.1 shows such a Gantt chart. It is generated by the simulation model by calling a library routine that we programmed for this purpose. It shows the loading of each machine and the stages from which they originate. The Gantt chart for the next period can be generated in order to control the progress of the jobs.

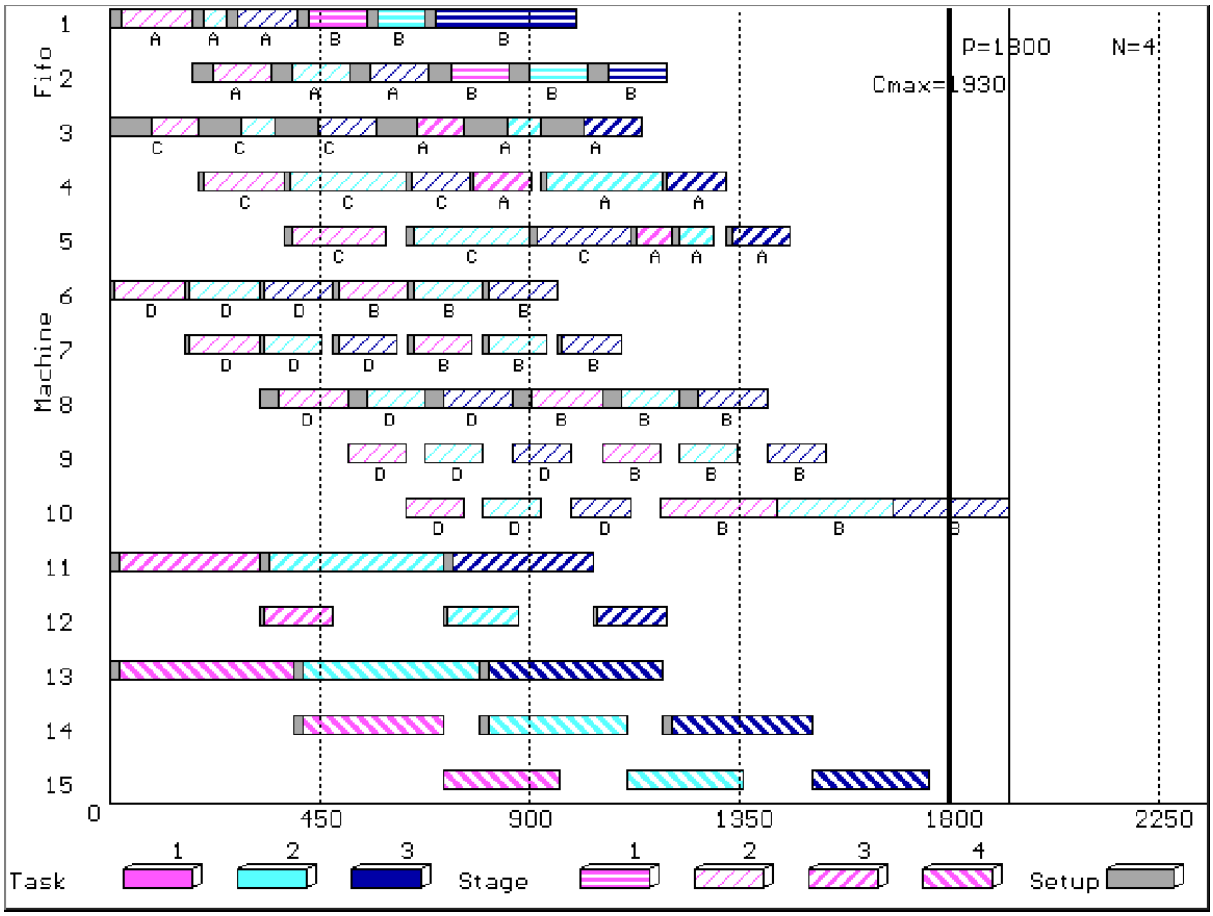



**Figure F.1 Gantt chart used for validation of the simulation model**

The next validation tool that we used was discussing the outcomes with colleagues. The Gantt charts helped in these discussions. Keep in mind that the schedules need not be realistic in a sense that the schedule should be usable in practice. The model does not consider trade-offs that are important in practice, such as delaying the start of new work just before the end of the day, minimal time lags between successive operations, and so on. However, the model generates a non-delay schedule that can function as an indication of a real lowerbound on the make span performance of the system.

Finally, we have compared our simulation model with the description of the production system that was simulated by Steele, Berry and Chapman (1995), and that was also used in Steele and Malhotra (1997). We have also analysed their results for comparable situations (N=4, weak cell position=5, period length = 4 days), but our fundamental choice to model the basic unicycle PBC system that does not transfer work to the next period but finishes it within overtime makes it difficult to compare the results. However, the general pattern of their results can also be found in our experiments.

An important aspect of validation is the length of the warming-up period. We distinguish between the warming-up during a simulated PBC period and the length of the start-up interval before the outcomes of the simulation study are reported. The start-up interval has to be such that the system is in steady state. This state is reached by our system after N production periods, as the final stage N releases work to the system that was arrived at the system just before the first period. Therefore, the start-up period has to be greater than N+1. We have taken N+3 as start-up period and apply a test in the program whether this state has been reached. We do not consider a warming-up period, as we examine a terminating simulation system. The behaviour at both the start and the end of the period can be very important for the outcomes. The warming-up during a period is therefore important, as we are interested in the start/finish effect.

# Appendix G. Modifications in progressive search heuristic

Chapter Seven describes the application of the progressive search heuristic in determining a configuration for a PBC system. The progressive search heuristic of Chapter Five requires some modifications in order to be able to determine a PBC configuration for the product structure of production situation I. Other modifications are required in order to test the proposed configuration in the simulation model of Chapter Six. Recall that the progressive search heuristic has been applied on a set of problems with a relatively simple structure, while production situation I resembles a more realistic situation.

Section G.I discusses modifications in the lowerbounds on period length P that the progressive search heuristic applies. Section G.II describes differences in modelling completion time. Section G.III presents modifications because of differences in the cost structure of both models.

## G.I. Modifications in lowerbounds on P

The progressive search heuristic applies two lowerbounds on the period length P: a longest-path oriented lowerbound that determines P according to the product with the longest throughput time, and a load oriented lowerbound that determines the minimal period length for which enough time in a period remains to complete the required operations. The following problems occur which make modifications in the heuristic necessary:

The *longest-path oriented lowerbound* in the heuristic presumes a linear product structure. However, the product structure in production situation I is not linear but convergent, as is quite normal in assembly operations. A convergent product structure consists of multiple paths that all lead to the same final operations. Figure G.1 shows such a convergent[1] product structure.

[1] This notion of convergence can with the use of graph theory be stated more formally as: The directed acyclic graph G containing vertices X (operations) and arcs A (finish to start precedence relationships between two different vertices) is convergent if the outdegree of each vertex is one except for the only sink vertex that has no successor at all (outdegree = zero), while the indegree of all vertices is at least one except for the various source vertices that have no predecessors at all (indegree = zero). A path in a convergent directed acyclic graph consists of a sequence of adjacent vertices. See for an introduction of graph theory Minieka (1978).

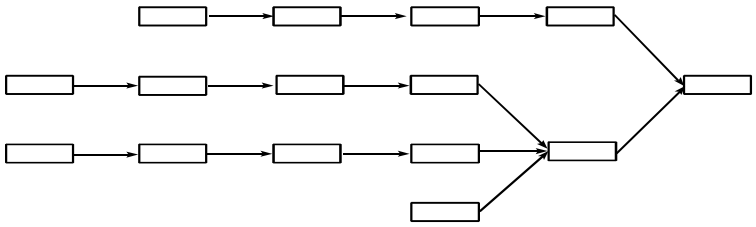


**Figure G.1 Convergent product structure**

We are interested in the *longest path* in the convergent product structure of each product in production situation I. However, which path will be longest may depend on the subbatch strategy, on the processing time and set-up time of each operation, on the volume that has to be produced in a period and hence on the period length.

We have modified the progressive search heuristic such that we added an initial step in which we determine the operations that are on the longest path in case no extra transfer batches are used. These are the only operations we use in our method for determining the period length P, number of stages N, and subbatch strategy[2]. The main problem with this approach is the effect of a proposed subbatch strategy, which indeed might shorten this path, but may have no effect at all on the other paths in the product structure. Often this is no problem, as other paths will not require overlapping production at all if they carefully distribute the co-ordination efforts over the various paths. However, the proposed subbatch strategy may lead to another path becoming longest path. This is not visible for the progressive search heuristic if it is modified as proposed. Further improvements of the modification would have been possible, but were not necessary for achieving our research objective. Therefore, we restricted ourselves to this modification and checked the data of production situation I whether the other paths in the product structure could be shortened. We conclude that there are no problems with respect to the possibility of reducing these other paths through an appropriate subbatch strategy if required by the heuristic.

The costs of shortening these parallel paths are not included in the total costs that our progressive search heuristic reports for a specific configuration of the PBC system. Shorter period lengths or lower number of stages will therefore seem to be more attractive to the progressive search heuristic than we may expect when we test these configurations with the original product structures in our simulation model. In order to count for this effect, we have introduced higher transfer batch costs and holding costs per operation in the progressive search heuristic. This compensates for the smaller number of operations that it considers.

[2] For the data of production situation I, the selection of operations that belong to the longest path was not influenced by the volume that had to be produced in a period. Neither were there differences between the products with respect to the machines that were involved in performing the longest path operations.

The *load oriented lowerbound* in the heuristic also presumes a linear product structure. Operations that require capacity at a specific resource are on the routing of a product and this routing has a linear structure. However, in production situation I operations on non-critical paths do also require set-up time and processing time on the same resources as operations on the longest path. Therefore, we modified the progressive search heuristic to include the required capacity of these operations in determining the load oriented lowerbound. This results in a more realistic minimal period length determination.

If we are able to apply a modified progressive search heuristic on the data of production situation I, we will obtain a period length P, number of stages N, a subbatch policy, and the expected total costs that will be incurred with such a PBC configuration. These parameters of the PBC system configuration cannot directly be imported into the simulation model. First, we have to assure that the data that is used in both models is in accordance.

Recall that both models are being designed with a different purpose. The progressive search heuristic had to find an approximate solution for the mathematical model presented in Chapter Five, which included a variable subbatching strategy and no predetermined allocation of operations to stages. The simulation model is being designed for evaluating several combinations of N and P and other PBC design parameters, such as the equal subbatch strategy, stage co-ordination policy, and cell priority rules, given an allocation of operations to stages. It shows the significance of main and interaction effects of these design factors on PBC system performance. In order to show significance of these effects, experiments with a variable subbatch strategy were not necessary, and they would have increased the amount of calculations strongly. The simulation model is hence not able to simulate configurations with a variable subbatch strategy.

The attempt to use the results of one model as input in the other makes it further necessary to explore the differences between both models in terms of cost structure and behaviour with respect to the sequencing of activities.

## G.II. Modifications in modelling completion time

The progressive search heuristic models the expected completion time of a product according to the formula $TT_h$ (Expression (12) in Chapter Five), corrected for a machine interference parameter MI (Expression (10)). Two main modifications in the heuristic were necessary:

First, Expression (12) in Chapter Five provides a formula for $TT_h$/MI that allows each product to have a different number of subbatches per operation. The simulation results for production situation I that we have presented in Chapter Six are all based on an equal number of subbatches strategy: equal for all products and all operations. We would like to compare the results of the configuration proposed by the progressive search heuristic with the results of the

corresponding original PBC configuration from Chapter Six. We simulate the system with the new P and N for all equal subbatch strategies that can be deduced from the specified subbatch strategy. As we used a maximum of four subbatches in the simulation experiments, we modified the progressive search heuristic such that each operation could have no more than four subbatches.

Second, the parameter MI has to be set in order to obtain a configuration of the PBC system. We have taken into account that this parameter needs to be smaller than one in order to accomplish for three important modelling differences between both models:
First, the progressive search heuristic assumes that the set-up of the machine that is required for the next operation on the path can be performed in parallel with processing the first subbatch at the preceding operation. The simulation model assumes that a machine decides about the required set-up as soon as the first transfer batch arrives. This lengthens the required throughput time in the simulation model and can be corrected with a parameter choice MI < 1 in the progressive search heuristic.

Second, the simulation model applies a predetermined allocation of operations to stages that may differ from the allocation that best suites the proposed configuration of the PBC system originating from the progressive search heuristic. The results of testing the proposed configuration will therefore be less positive than expected in the heuristic in terms of required amount of overtime work. A parameter choice of MI < 1 may compensate for this effect.

Finally, the nearer the proposed period length is to one of the lowerbounds in the progressive search heuristic, the more sensitive it is to the assumptions behind these lowerbounds. If the period length is near a lowerbound, we cannot expect that it will be sufficiently long to avoid overtime work. For the path oriented lowerbound, we can choose a machine interference correction factor MI < 1 as discussed before. For the load oriented lowerbound in the progressive search heuristic, we can apply the same reasoning. This lowerbound assumes that a resource never has to wait on the arrival of a batch that requires capacity of this resource. If preceding operations at other resources have to be performed in the same period, a start/finish delay is introduced. This waiting time can be accomplished for in the load oriented lowerbound by requiring that the product of the period length P and a correction factor has to exceed the load oriented lowerbound. Both lowerbounds use the same value for the correction factor.

## G.III. Modifications in cost structure

The progressive search heuristic estimates the total costs *per year* for various configurations of the PBC system. As we used in our simulation experiments a fixed throughput time of T=7200 minutes and computed costs over this period, we have modified the progressive search heuristic such that it expresses the cost of a solution per T=7200 minutes[3]. The value for P is now also expressed as a fraction of T=7200.

The progressive search heuristic assumes that total costs consist of three factors: echelon holding costs, set-up costs, and transfer costs. The costs of overtime are not taken into account, as it assumes that no overtime will occur with the configuration that is being determined. In the simulation study, we have used a finite overtime cost factor. The progressive search heuristic is not modified for this, so it will not try to use overtime work because of the smaller cost factor.
The cost structure for transfer batches is identical in both models and the same holds true for the set-up costs. However, the calculation of the costs of holding inventory in the progressive search heuristic is very different from the calculation in the simulation model. The heuristic assumes that all material on average is available half way the first stage and stays therefore N periods of length P in the system. It computes the echelon holding costs over this period. The simulation model considers more details on the exact timing of operations and on the stage in which bought material is required. Simulated holding costs depend therefore on the allocation of operations to the stages and the timing of these operations. A higher number of stages in the simulation generally causes an echelon holding cost decrease even if the product of N·P remains equal. This is contrary to the results of the heuristic, which will report identical holding costs as long as N·P remains constant. As these holding cost differences cannot be corrected through a modification of the progressive search heuristic, we have determined values for the holding cost factor that result in almost equal total holding costs for a random selection of cases.

Having described all these modifications, we return to discussion on the applicability of the progressive search heuristic to the problem of determining a configuration for a PBC system in production situation I. The two models are being developed with a different purpose and therefore do not resemble the same structure. We have proposed several modifications of both the heuristic and the input (cost) data in order to be able to use the results of one model as input in the other model and to compare the results with other simulated configurations. With these modifications, we test the proposed PBC configuration resulting from the progressive search heuristic. Section § 7.1 describes the results of these tests.

[3] If we simulate PBC configurations with N·P < 7200 minutes, we recalculate the reported costs for a standard period of 7200 minutes.